

Ansible AWX





Ansible AWX

-
- An Overview of Ansible AWX/Tower.
 - Features of Ansible AWX.
 - Features of Ansible Tower.
 - Similarities Between Ansible AWX and Ansible Tower.
 - Key Differences Between Ansible AWX and Ansible Tower.
 - When to Use Ansible AWX vs. Ansible Tower.

Ansible AWX and Ansible Tower

HIGHLIGHTS

- **Ansible AWX** is the open-source version of **Ansible Tower**, which is a web-based interface for managing Ansible automation.
- Both tools are used for automating IT tasks like configuration management, application deployment, and task automation.
- They offer a more user-friendly way to interact with Ansible, especially for larger teams and enterprise environments.

Ansible AWX and Ansible Tower

ANSIBLE AWX

- **Free and Open-Source:** AWX is the free version, available to anyone who wants to use it.
- **Community Support:** It is supported by the community, which means there is no official help if you run into problems.
- **Basic Features:** It has all the basic features to manage automation tasks but doesn't have some advanced features like security tools or reporting.

ANSIBLE TOWER

- **Paid Version:** Tower is the commercial (paid) version of AWX, with extra features that businesses need.
- **Official Support:** Tower comes with customer support from the company behind Ansible (Red Hat), which can help if you run into issues.
- **Extra Features:** Tower has more advanced features like security tools, better reporting, and the ability to handle larger and more complex environments.

Ansible AWX and Ansible Tower

SIMILARITIES

- **Automation Management:** Both let you run and manage automation tasks through a user-friendly web interface.
- **Job Scheduling:** Both allow you to schedule tasks and control who can run them using roles (permissions).
- **API:** Both have an API (a way for other software to interact with them).

Ansible AWX and Ansible Tower

KEY DIFFERENCES

- **Cost:** AWX is open-source and free to use, whereas Ansible Tower requires a commercial license, with pricing based on the number of nodes being managed.
- **Support:** Ansible Tower comes with official support, while AWX relies on community support.
- **Enterprise Features:** Tower offers advanced features such as clustering, integration with enterprise security systems (e.g., LDAP, SAML), and audit logging that are not present in AWX.
- If you're just starting or have a smaller setup, AWX might be enough. If you're working in a big company and need more advanced features and support, Tower would be a better choice.



Ansible AWX

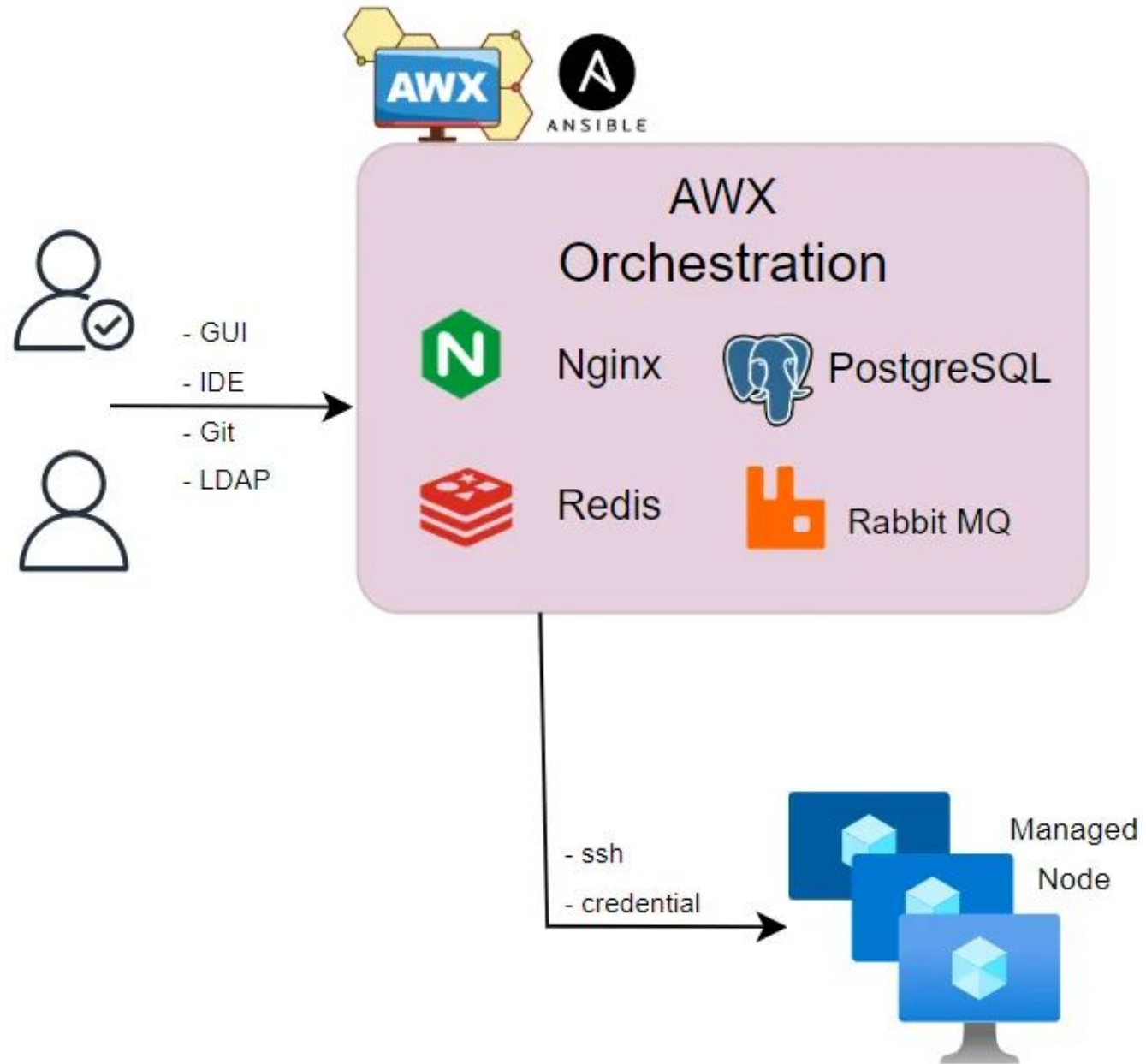


Ansible AWX

Ansible AWX provides the following features:

- Ansible Project Management and Host Management
- Provisioning and Configuration Management
- CD (Continuous Delivery) & Workflow Template
- Centralized logging/audit
- Authentication integration (LDAP, SAML, GitHub, Google, Azure AD)
- Visualized web dashboard
- High availability

Ansible AWX



Ansible AWX Components

The components of AWX include:

Nginx:

- This is a web server that acts as a reverse proxy, loadbalancer, mail proxy, and HTTP cache.
- In AWX, it is responsible for routing client requests to the appropriate services and delivering responses from those services to the client.

PostgreSQL:

- This is the main database used by AWX. It stores information such as AWX's settings, user info, and job results.
- PostgreSQL provides features such as transactions and recovery, subqueries, triggers, and replication to support the reliability and scalability of AWX.

Ansible AWX Components

Redis:

- This is an in-memory data structure store that is used as a database, cache, message broker, and so on.
- In AWX, it is responsible for fetching and processing messages from the job queue.
- It also improves performance by caching state information, job results, and etc.

RabbitMQ:

- This is a message broker, which is responsible for asynchronous message delivery between AWX components.
- In AWX, it's responsible for enqueueing job requests and letting Redis handle them.

Ansible AWX Setup

- **AWX installs on top of Docker on Linux OX up to version 17 (k8s is also possible).**
- **From version 18 onwards, AWX defaults to installing on top of Kubernetes (no Docker support).**
- **References:**

<https://github.com/Ansible/awx>

<https://github.com/Ansible/awx/blob/devel/INSTALL.md>

Ansible AWX 17 Installation steps

- You have to turn on the virtualenv venv and proceed.

If you have venv enabled, you will need to replace the `ansible_python_interpreter` path in `~/awx/installer/inventory` with the `venv python` path. Installation methods may vary from environment to environment.

- Install Ansible

```
$ yum install ansible -y
```

- Install pip

```
$ yum isntall python3-pip
```

- Install Docker

<https://github.com/docker/docker-py/issues/3194>

```
(venv)$ pip install docker==6.1.3
```

- Install Docker Compose

```
(venv)$ pip install docker-compose
```

Ansible AWX 17 Installation steps

- Install Git & AWX 17.1.0

```
$ yum -y install git
```

```
$ git clone --branch 17.1.0 --single-branch https://github.com/ansible/awx.git
```

- Configuration

```
$ cd ~/awx/installer    # It is related to my AWX path
```

```
$ vi inventory    # This file contains the env information required to install AWX
```

```
# admin_password=password <-- Uncomment : change password you want
```

```
# project_data_dir=/var/lib/awx/projects <-- Uncomment
```

```
$ yum -y install libselinux-python3
```

```
$ ln -s /usr/local/bin/docker-compose /usr/bin/docker-compose
```

```
$ ansible-playbook -i inventory install.yml -b
```

- Confirm

```
$ docker ps
```

Ansible AWX 17 Installation steps

The basic workflow between AWX components is roughly as follows

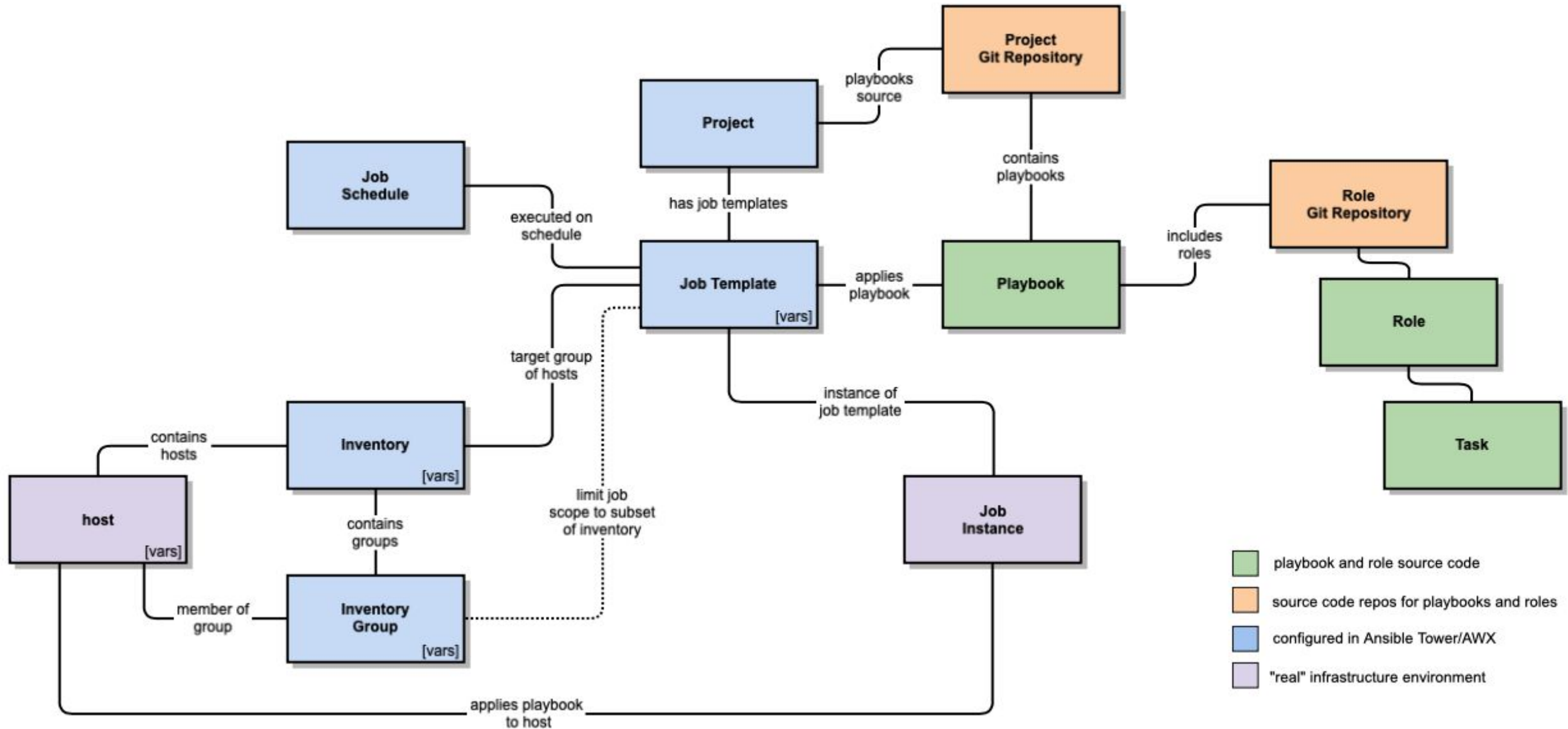
1. A user requests to run an Ansible Playbook through the **AWX web interface**.
2. The request is passed through **Nginx** to the **AWX API server**.
3. The **AWX API** server passes the request to **RabbitMQ**, which adds it to the **job queue**.
4. **Redis** fetches the job from **RabbitMQ's job queue** and processes it.
5. The **job's execution** results are stored in a **PostgreSQL database** and **cached** by **Redis** if necessary.
6. The user can view the job's execution **results** through the **AWX web interface**.



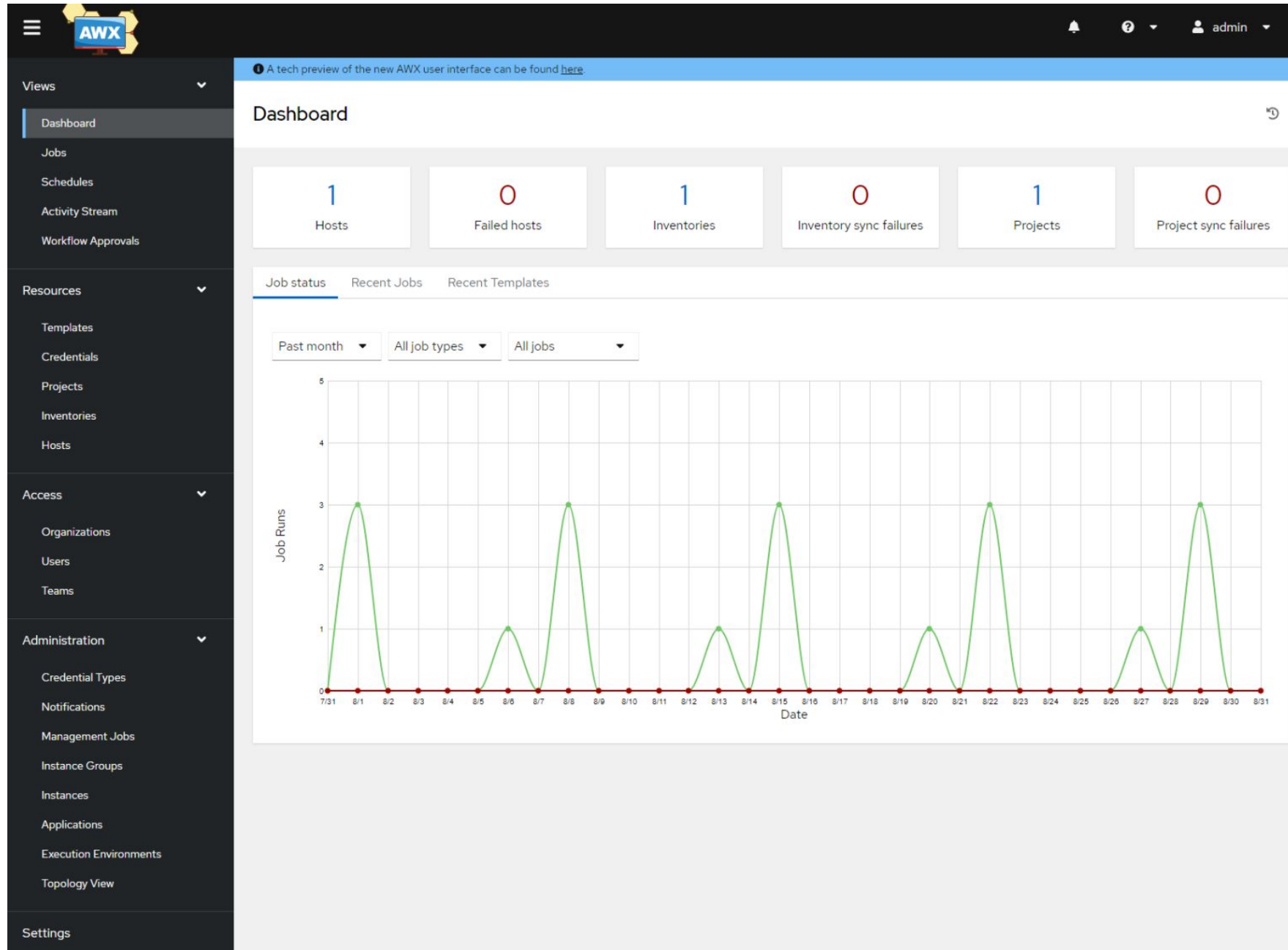
Basic usage of Ansible AWX

-
- Create an Organization through the AWX web interface.
 - Create a Project and assign it to the created organization.
 - Set Up Credentials, including the private key and other required details.
 - Create an Inventory, add hosts to it, and test connectivity using the ping module.
 - Create and Execute a Job Template to run your playbook or tasks.

AWX components overview

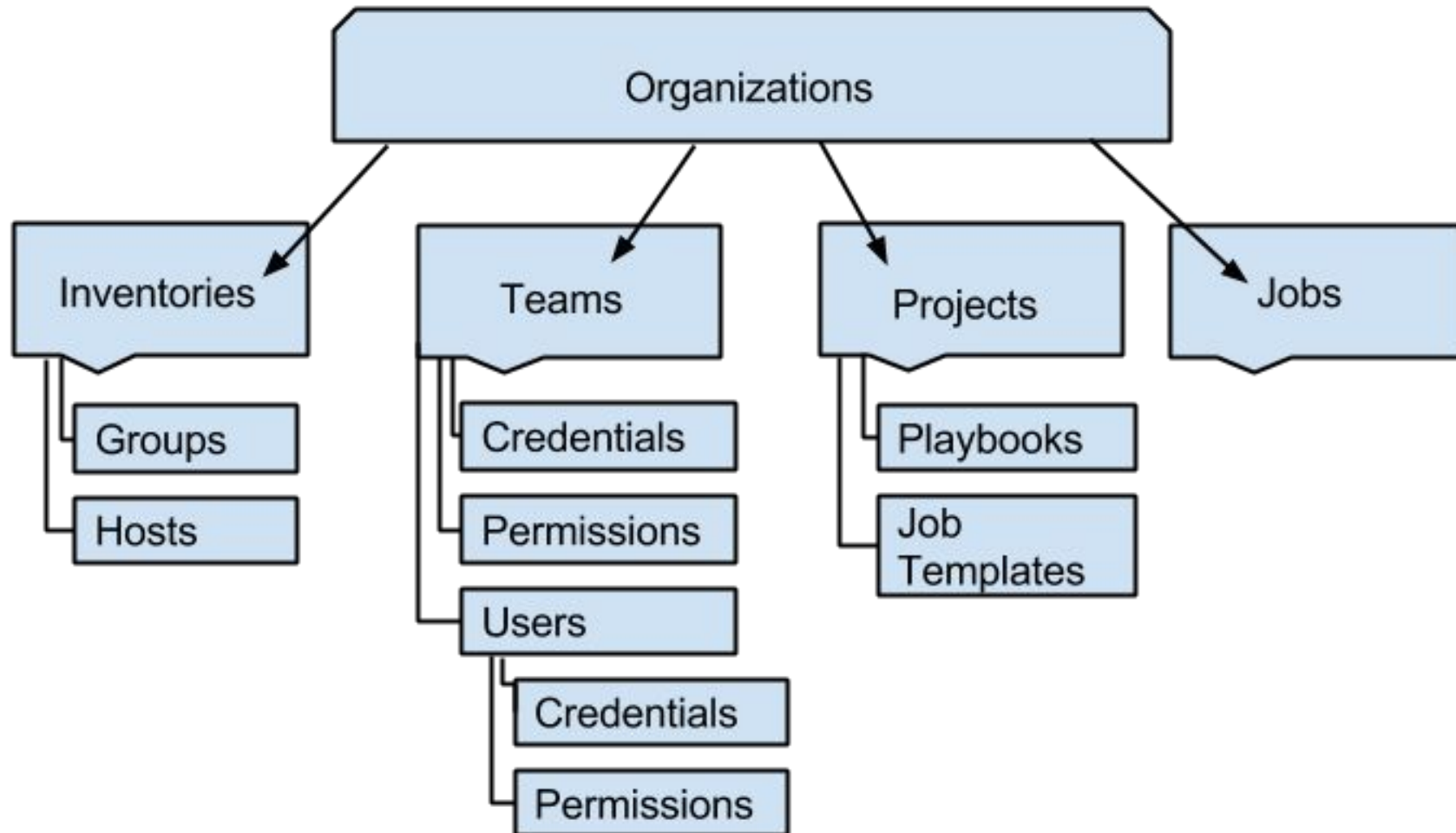


AWX Dashboard



AWX Organizations

An Organization is a logical collection of Users, Teams, Projects, and Inventories, and is the highest level in the AWX object hierarchy.



AWX Credentials

- Credentials are utilized for authentication when launching Jobs against machines, synchronizing with inventory sources, and importing project content from a version control system.
- The target can be a server, or it can be Github (or any other SCM solution).
Typically, you would register an SSH private key or password to access the server so that it is available to authorized users without exposing its contents.

AWX Credentials

- Credential Types:

- Amazon Web Services
- AWS Secrets Manager
- Ansible Galaxy/Automation Hub API Token
- Centrify Vault Credential Provider Lookup
- Container Registry
- CyberArk Central Credential Provider Lookup
- CyberArk Conjur Secrets Manager Lookup
- GitHub Personal Access Token
- GitLab Personal Access Token
- Google Compute Engine
- GPG Public Key
- HashiCorp Vault Secret Lookup
- HashiCorp Vault Signed SSH
- Insights
- Machine
- Microsoft Azure Key Vault
- Microsoft Azure Resource Manager
- Network
- OpenShift or Kubernetes API Bearer Token
- OpenStack
- Red Hat Ansible Automation Platform
- Red Hat Satellite 6
- Red Hat Virtualization
- Source Control
- Terraform backend configuration
- Thycotic DevOps Secrets Vault
- Thycotic Secret Server
- Vault
- VMware vCenter

AWX Inventories

- A list of target servers to work with, either as a list of IPs or IP lookupable hostnames.
- You can list servers one by one, or you can manage them in groups, or you can include groups within groups.
- When using Ansible directly, you create and manage a hosts file, but AWX supports building lists using several sources.

AWX Projects

Projects in AWX are used to:

- Define the source of your Ansible playbooks and related files
- Connect to version control systems (like Git repositories) to fetch playbook content
- Manage playbook updates and syncing

Key aspects of projects include:



- They specify where AWX should look for playbooks, roles, and other Ansible content
- Projects can be linked to Git repositories for version control
- They allow for easy updating of playbook content by syncing with the source repository

AWX Templates

Ansible **AWX templates** also known as **job templates**, are configurations within the Ansible AWX platform that define how Ansible playbooks are executed. They include ***various parameters and settings*** like:

- **Inventory:** Specifies the hosts or groups of hosts that the playbook should target. AWX allows for dynamic inventory management, enabling the use of scripts or external sources to generate the inventory dynamically.
- **Credentials:** Defines the credentials needed to authenticate with the target hosts, such as SSH keys, usernames, and passwords.
- **Playbook:** Specifies the Ansible playbook that should be executed. This can either be a playbook stored within the AWX project or a playbook fetched from an external source, such as a Git repository.
- **Limit:** Optionally restricts the execution of the playbook to a subset of hosts within the inventory.
- **Extra Variables:** Allows passing additional variables to the playbook, which can be used to customize its behavior dynamically.
- **Verbosity Level:** Specifies the level of detail for the output generated during playbook execution.
- **Timeouts:** Defines the maximum duration allowed for playbook execution before it is considered failed.
- **Notification Settings:** Configures how notifications should be handled upon playbook completion, including email notifications or triggering webhooks.

AWX Users



Schedules

Activity Stream

Workflow Approvals

Resources

Templates

Credentials

Projects

Inventories

Hosts

Access

Organizations

Users

Teams

Administration

Credential Types

Notifications

Management Jobs

Instance Groups

Instances

Applications

Execution Environments

Topology View

Settings

Users

Create New User

First Name

Last Name

Email


myUser

Username *


Password *

Confirm Password *

myUser



.....




.....

User Type *

Organization *

Normal User



myOrg

Save

Cancel

AWX Instance Groups

- An instance group in AWX is a structure for managing the computing resource on a system used to process AWX jobs. In other words, an instance group represents a **collection of AWX instances** running **Ansible playbooks**.
- By managing multiple instances together as an instance group, you can balance the load of AWX jobs. This prevents the load from being concentrated on a single instance and improves the performance and reliability of the overall system.
- On the other hand, AWX's inventory is a structure that manages the hosts that are the target of Ansible jobs.

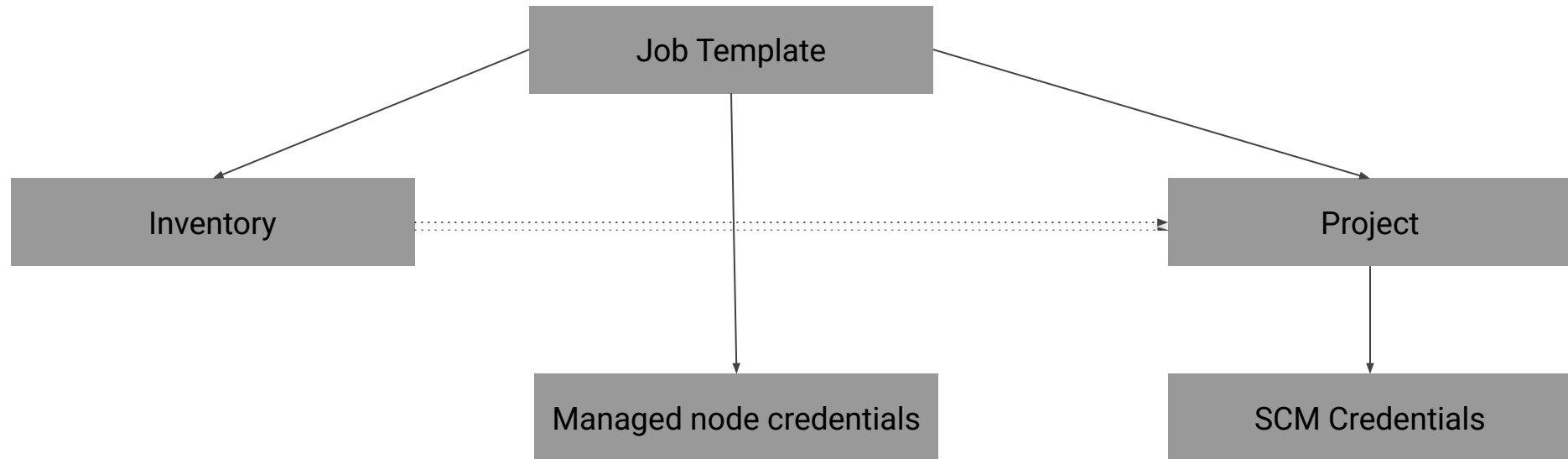
Ansible AWX Integration with Git

❑ Workflow Summary:

- ❑ **Step 1:** The playbook and inventory files are stored in a Git repository.
- ❑ **Step 2:** Ansible AWX fetches the configurations from the Git repository.
- ❑ **Step 3:** AWX connects to the target systems (Ubuntu and RHEL) via SSH.
- ❑ **Step 4:** The tasks in the playbook are executed on the target systems to automate the desired configurations.

Creating AWX resources

Prior to creating a job template a few resources that are linked by it must be created. The resources will be created bottom up, i. e. those located lower in the object graph will be created first.



First, the Git and managed node credentials must be added to AWX. After that a project is created, which references the Gitea credentials to access all Ansible files.

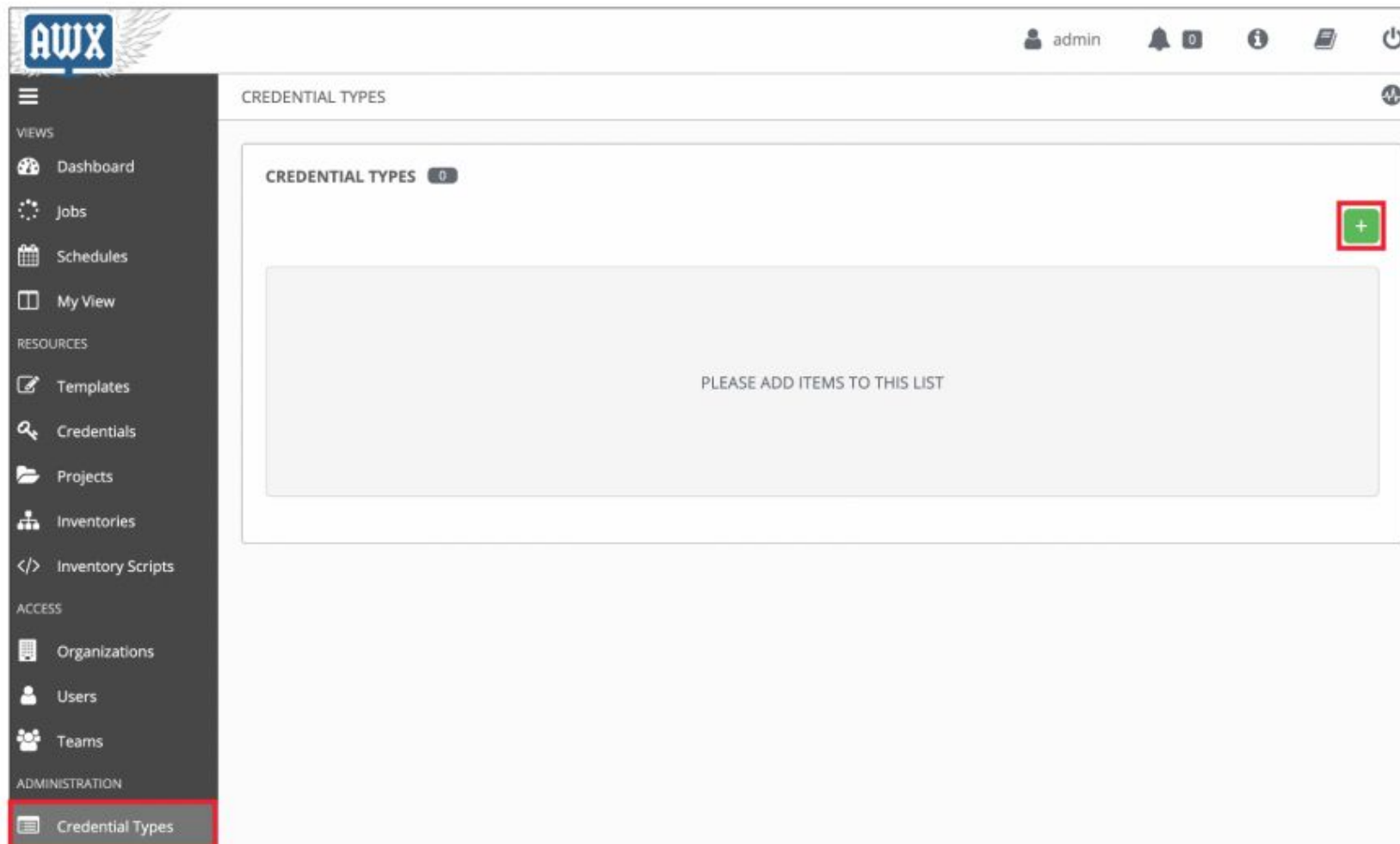
Next an inventory is created. While this inventory is based on the inventory file from Gitea, it will not reference the Gitea credentials, but rather use the project as a source.

Finally a job template is created, which links the project and inventory to a unit.

Create credential Types

Let say that you have a playbook containing variables that need to be defined with credentials for logging into the AWX REST API framework. In order to accomplish this, a custom credential type will need to be created.

First, click the Credential Types menu on the left, then click the + button:



Create credential Types

In the New Credential Type page. Enter the following into the required fields:

- **NAME:** Enter the name of the credential type. Here, it is named AWX REST API.
- **INPUT CONFIGURATION:** Ensure YAML is selected. Then copy and paste the following into the box:

```
fields:
- id: username
  type: string
  label: AWX Username
- id: password
  type: string
  label: AWX Password
  secret: true
required:
- username
- password
```

INJECTOR CONFIGURATION: Ensure YAML is selected. Then copy and paste the following into the box:

```
extra_vars:
  awxlogin_user: '{{username}}'
  awxlogin_pass: '{{password}}'
```

Create credential Types

NEW CREDENTIAL TYPE

* NAME

AWX REST API

DESCRIPTION

INPUT CONFIGURATION ?

YAMLJSON

```
1 fields:
2   - id: username
3     type: string
4     label: AWX Username
5   - id: password
6     type: string
7     label: AWX Password
```

INJECTOR CONFIGURATION ?

YAMLJSON

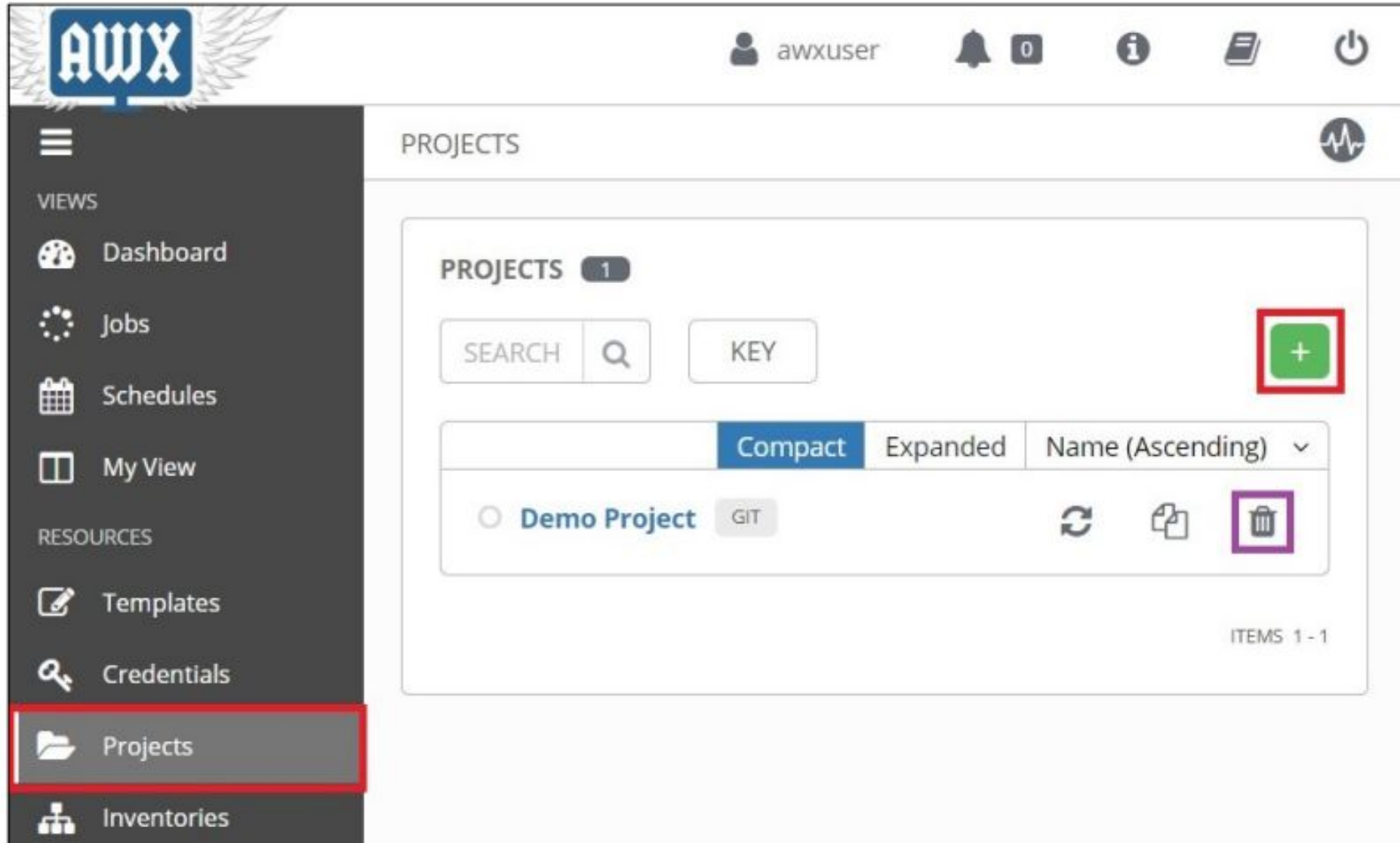
```
1 extra_vars:
2   awxlogin_user: '{{username}}'
3   awxlogin_pass: '{{password}}'
4
```

CANCEL

SAVE

Create credentials

Go to *Resources* → *credentials* in AWX and click the "+" button.



Create credentials

On the new page choose the newly created credential type. enter the username and the password required by the AWX Login credential type:

NEW CREDENTIAL

DETAILS

PERMISSIONS

* NAME ?

AWX Login

DESCRIPTION ?

ORGANIZATION ?

Q

SELECT AN ORGANIZAT

* CREDENTIAL TYPE ?

Q

AWX REST API

TYPE DETAILS

* AWX USERNAME

Q

awxuser

* AWX PASSWORD

Q

.....

CANCEL

SAVE

Create credentials

Save and create another credentials object, this time for the managed node(s)

Choose *Machine* for *credentials type*. Enter any information required, such as username and private key, and click *Save*.

CREDENTIALS / CREATE CREDENTIAL

NEW CREDENTIAL

DETAILS

PERMISSIONS

* NAME ?
ICX Login

DESCRIPTION ?

ORGANIZATION ?

Q SELECT AN ORGANIZA*

* CREDENTIAL TYPE ?

Q Machine

TYPE DETAILS

USERNAME

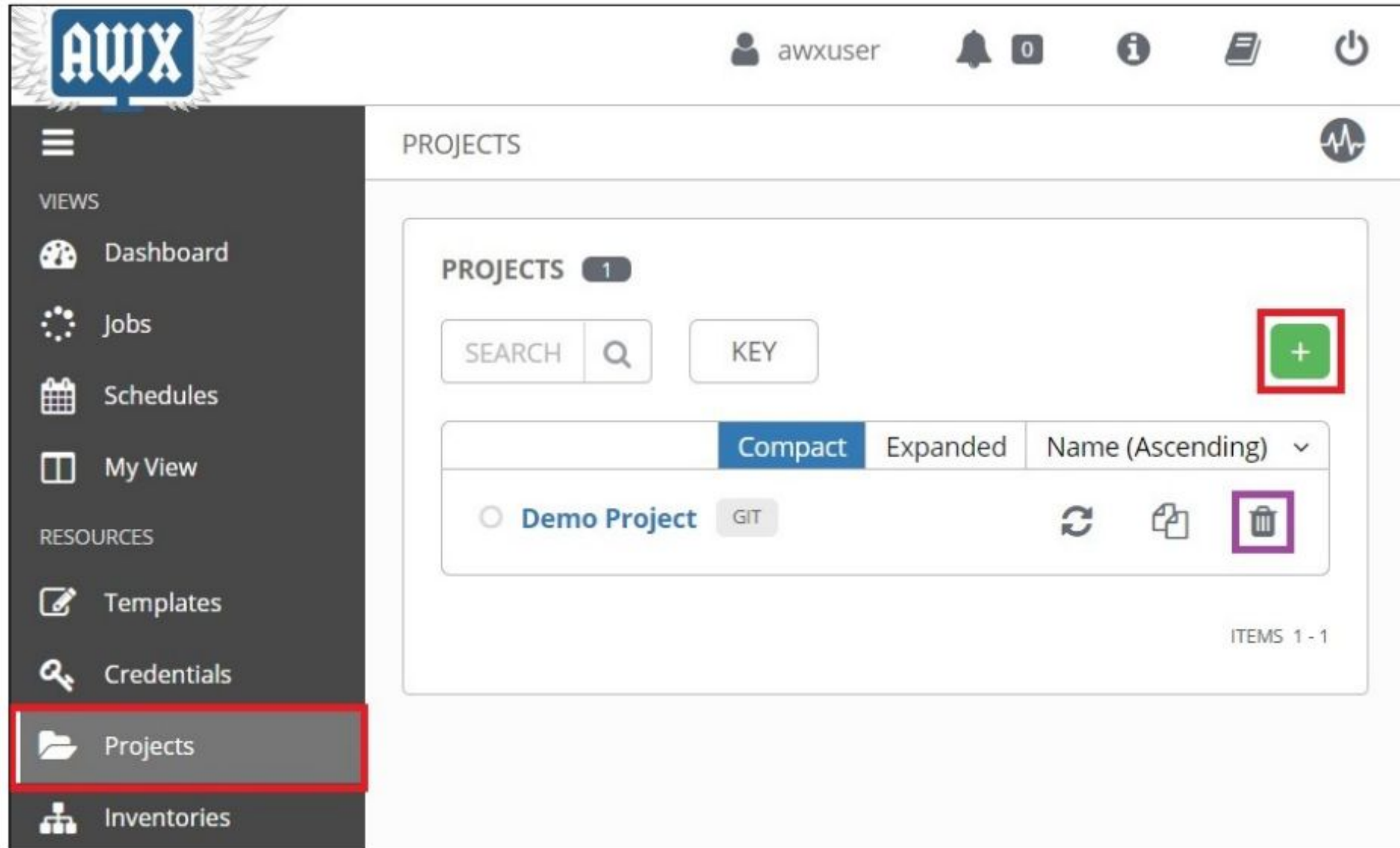
Q admin

PASSWORD ☐ Prompt on launch

Q

Create project

Go to Resources → Projects in AWX and click the "+" button.



Create project

On the new page choose **Git as SCM type** and enter the SCM URL. For SCM credential **use the previously created object**.

In SCM update options check clean and update revision on launch. (This will ensure every run will use the latest commit without local modifications. It will not update anything in the Git repository.) Confirm with Save.

NEW PROJECT

DETAILS

PERMISSIONS

JOB TEMPLATES

SCHEDULES

* NAME

AWX-Management

DESCRIPTION

* ORGANIZATION ?

Q Default

* SCM TYPE

Git

SOURCE DETAILS

* SCM URL ?

SCM BRANCH/TAG/COMMIT ?

SCM REFSPEC ?

SCM CREDENTIAL

Q

SCM UPDATE OPTIONS

- ☐ CLEAN ?
- ☐ DELETE ON UPDATE ?
- ☐ UPDATE REVISION ON LAUNCH ?
- ☐ ALLOW BRANCH OVERRIDE ?

CANCEL



SAVE

Create project- AWX Git access





Note that saving the project will fetch an initial version from your Git repository. The following steps rely on files found in this project, such as inventory files.


If there are no commits yet or any error occurred during retrieving the repository contents, a manual sync will be required before continuing. To retry click the circular arrows icon in the project overview or rocket icon in the details view.

PROJECTS 2

Compact Expanded Name (Ascending) ▾

 **AWX-Management** GIT   

☐ **ICX Test** MANUAL 

ITEMS 1 - 2

Create inventory

Go to Resources → Inventories in AWX and click the "+" button. Select Inventory

The screenshot displays the AWX web interface. On the left sidebar, the 'Inventories' menu item is highlighted with a red box. The main content area shows the 'INVENTORIES' tab selected. In the top right corner of the main content area, a green '+' button is highlighted with a red box. A dropdown menu is open from this button, showing 'Inventory' and 'Smart Inventory' options, with 'Inventory' selected and highlighted by a red box. Below the dropdown, a table lists the inventory items:

NAME	TYPE	ORGANIZATION
ICX Switches	Inventory	Default

The table has columns for NAME, TYPE, and ORGANIZATION. The first row shows 'ICX Switches' as the name, 'Inventory' as the type, and 'Default' as the organization. The table also includes icons for editing, copying, and deleting each item. At the bottom right of the table, it says 'ITEMS 1 - 1'.

Create inventory

Go to Resources → Inventories in AWX and click the "+" button. Select Inventory

NEW INVENTORY

DETAILS

PERMISSIONS

GROUPS

HOSTS

SOURCES

COMPLETED JOBS

★ NAME

AWX Server

DESCRIPTION

★ ORGANIZATION

Q

Default

INSIGHTS CREDENTIAL

Q

INSTANCE GROUPS ?

Q

VARIABLES ?

YAML

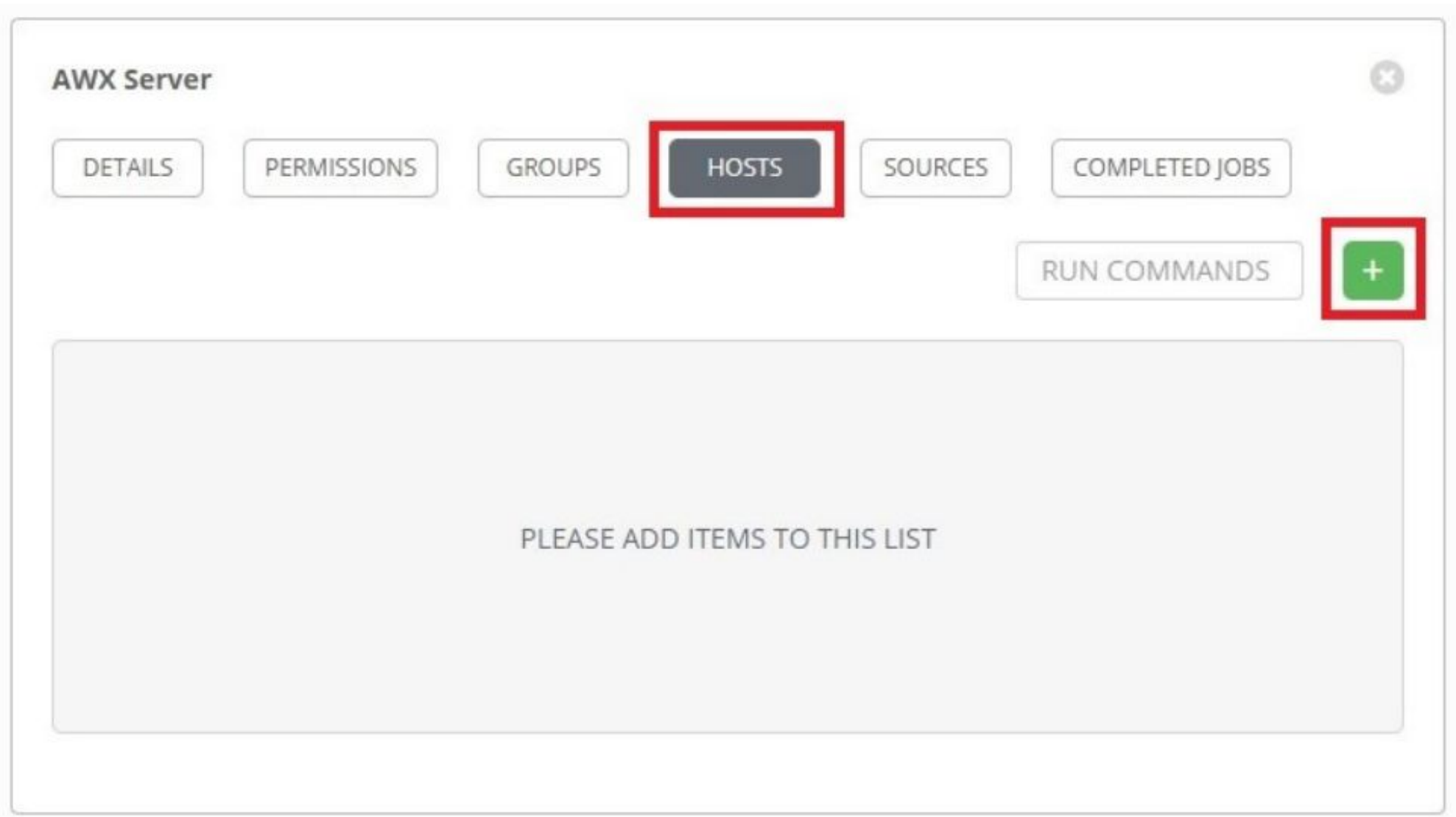
JSON

EXPAND

1

Create inventory - ADDING HOST

Click the HOSTS tab, then click the + button:



Create inventory - ADDING HOST

This opens the Create Host page. Complete the following fields:

- HOST NAME: Enter the hostname of the AWX server. Here it is called awxserver.
- VARIABLES: Type in ansible_host: followed by the IP address of the AWX server.

The screenshot shows the 'Create Host' page in AWX. At the top, the host name 'awxserver' is displayed with a toggle switch. Below this, there are four tabs: 'DETAILS', 'FACTS', 'GROUPS', and 'COMPLETED JOBS'. The 'DETAILS' tab is selected. In the 'HOST NAME' field, the text 'awxserver' is entered and highlighted with a red box. The 'DESCRIPTION' field is empty. Below the 'HOST NAME' field, there is a 'VARIABLES' section with a toggle switch set to 'YAML'. The 'VARIABLES' field contains the text '1 ansible_host: 134.242.138.80', which is also highlighted with a red box. At the bottom right, there are two buttons: 'CANCEL' and 'SAVE'. The 'SAVE' button is highlighted with a red box.

Create inventory

It is recommended to link inventory files from your project in Git. You may choose a different type of inventory and / or add hosts manually in AWX.

Enter a name and the location of the inventory file.

AWX inventory file:

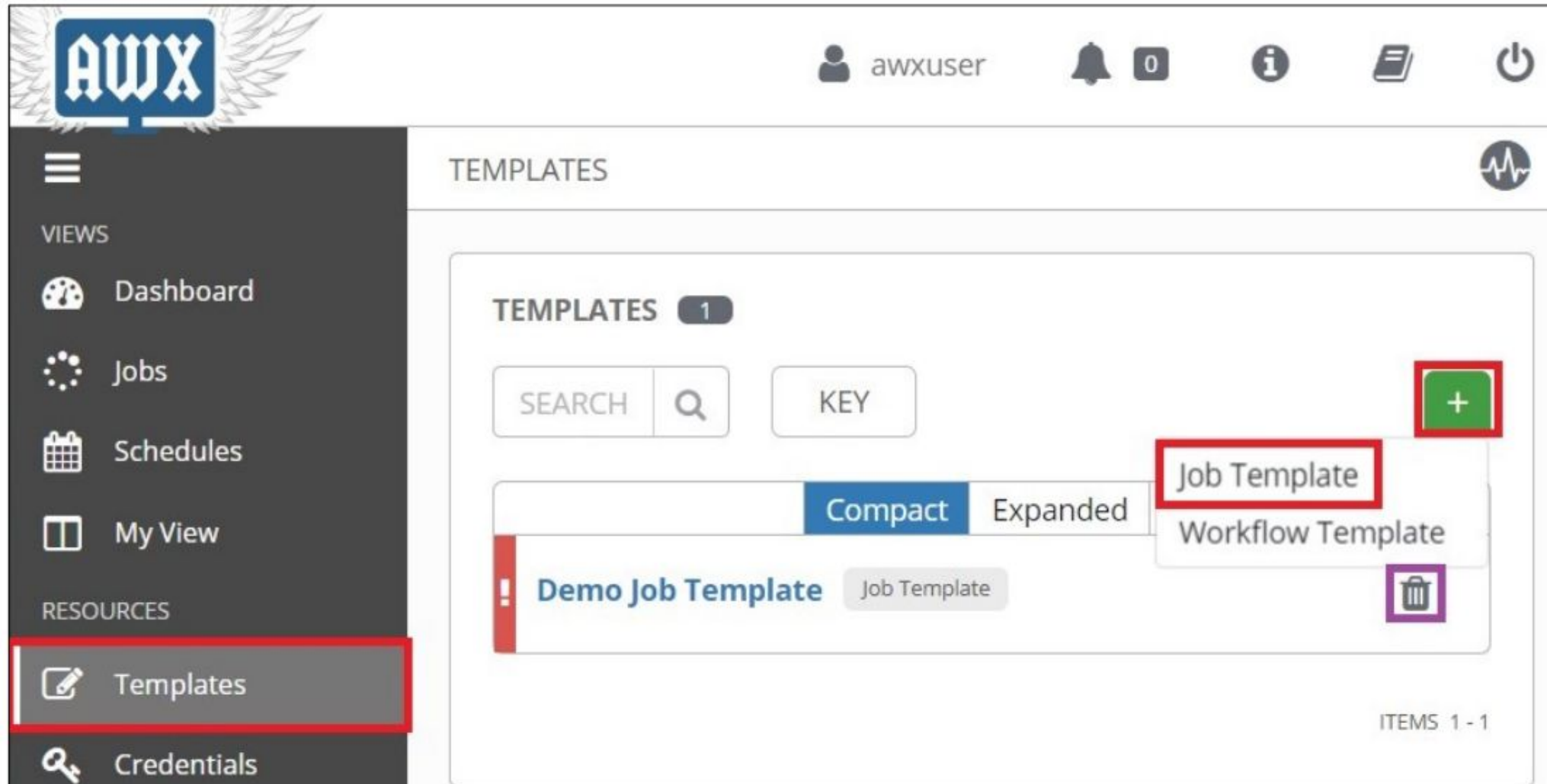
- You should specify the path to the inventory file instead of using the default / (project root), which scans the project root for INI- and YAML-style inventory files.
- Inventory files written in YAML and their **group_vars** may not be properly recognized if you use / (project root). Even if the hosts are found in the YAML-style inventory, **group_vars** are not, leading to unset variables. This is probably caused by a bug. Note: the path must not start with a "/".
- Check the following options:
Overwrite, Overwrite variables and Update on project update.

The option *Update on launch* is not available when using *Sourced from a project*. Confirm with *Save*.

Create job template

The last step before launching is to create the job template, which simply bundles all resources created so far.

Go to Resources → Templates in AWX and click the "+" button. Select Job Template.



Create job template

On the new page enter a name for the job template and choose Run as job type. Then fill in the form fields to point to the resources created so far:

- Inventory
- Project
- Playbook (read from project)
- Credentials

The screenshot shows the 'NEW JOB TEMPLATE' form with the following fields and values highlighted by red boxes:

- * NAME:** ICX Facts
- * JOB TYPE:** Run
- * INVENTORY:** ICX
- * PROJECT:** ICX Test
- * PLAYBOOK:** pbfacts.yml
- CREDENTIALS:** ICX Login

Other fields include DESCRIPTION, FORKS (0), and LIMIT, which are currently empty. There are also checkboxes for 'PROMPT ON LAUNCH' next to the Job Type, Inventory, and Credentials fields.

Create job template

Handling multiple playbooks:

If your project uses multiple playbooks for different tasks, such as awx-thales features build.yml and deploy.yml, you will have to create a second job template which points to the other playbook(s).

Alternatively, if you are using one playbook with tags for flow control, you will have to create a second job template which lists the desired tags (Job Tags or Skip Tags).

Running an AWX template

To run a job from a job template click the **rocket** icon in the list view or Launch in the detail view. If the job template contains a survey, enter the desired values and click Next and Launch.

The screenshot shows the AWX web interface. The top navigation bar includes the AWX logo, a user profile for 'admin', a notification bell with '0' alerts, an information icon, a document icon, and a power icon. The left sidebar contains a menu with 'VIEWS' (Dashboard, Jobs, Schedules, My View) and 'RESOURCES' (Templates, Credentials, Projects, Inventories, Inventory Scripts). The 'Templates' resource is highlighted with a red box. The main content area is titled 'TEMPLATES' and shows a list of 3 templates. The list has a search bar, a 'KEY' button, and a green '+' button. The list is displayed in 'Compact' view, sorted by 'Name (Ascending)'. The first template, 'AWX-Template-Management', is highlighted with a red box around its rocket icon. The second template, 'ICX Facts', and the third, 'ICX VLAN', also have rocket icons. The bottom right corner shows 'ITEMS 1 - 3'.

TEMPLATES	3			
SEARCH	Q	KEY	+	
		Compact	Expanded	Name (Ascending) v
AWX-Template-Management	Job Template			
ICX Facts	Job Template			
ICX VLAN	Job Template			

ITEMS 1 - 3

Running an AWX template

Once launched the job's details are displayed, allowing you to follow the output from ansible-playbook.

DETAILS

STATUS

Successful

STARTED

5/9/2021 9:04:28 AM

FINISHED

5/9/2021 9:04:44 AM

JOB TEMPLATE

AWX-Template-Management

JOB TYPE

Run

LAUNCHED BY

admin

INVENTORY

AWX Server

PROJECT

AWX-Management

REVISION

23b2c31

PLAYBOOK

AWX-Template-Management.yml

CREDENTIAL

AWX Login

ENVIRONMENT

/var/lib/awx/venv/ansible

EXECUTION NODE

awx

INSTANCE GROUP

tower

EXTRA VARIABLES

YAML JSON EXPAND

1

AWX-Template-Management

PLAYS 1 TASKS 17 HOSTS 1 ELAPSED 00:00:15

SEARCH Q KEY

35 TASK [self_update : Get Credentials for T 09:04:36

his Template] *****

36 ok: [awxserver]

37

38 TASK [self_update : Store Credentials fro 09:04:37

m This Template] *****

39 ok: [awxserver] => (item=0)

40

41 TASK [self_update : Add Credentials to 'I 09:04:38

mage-Management' Template] *****

42 ok: [awxserver] => (item=0)

43

44 TASK [self_update : Create New Survey Pro 09:04:39

mpt Payload] *****

45 ok: [awxserver]

46

47 TASK [self_update : Create Survey Prompt] 09:04:39

48 ok: [awxserver] => (item={'JobId': 13, 'SurveyP

rompt': {'name': '', 'description': '', 'spec':

[{'question_name': 'Template management functio

n:', 'required': False, 'type': 'multiplechoic

AWX-Template-Management

PLAYS 1 TASKS 17 HOSTS 1 ELAPSED 00:00:15

SEARCH Q KEY

54 ext , variable : prompt_ctrip_filepath }}}}

55 TASK [self_update : Set AWX Management Ve 09:04:43

rsion] *****

56 ok: [awxserver]

57

58 TASK [self_update : Update Message] ***** 09:04:43

59 ok: [awxserver] => {

60 "msg": [

61 "Template updated to version 1.

4.",

62 "Relaunch template to continue."

63]

64 }

65

66 PLAY RECAP ***** 09:04:44

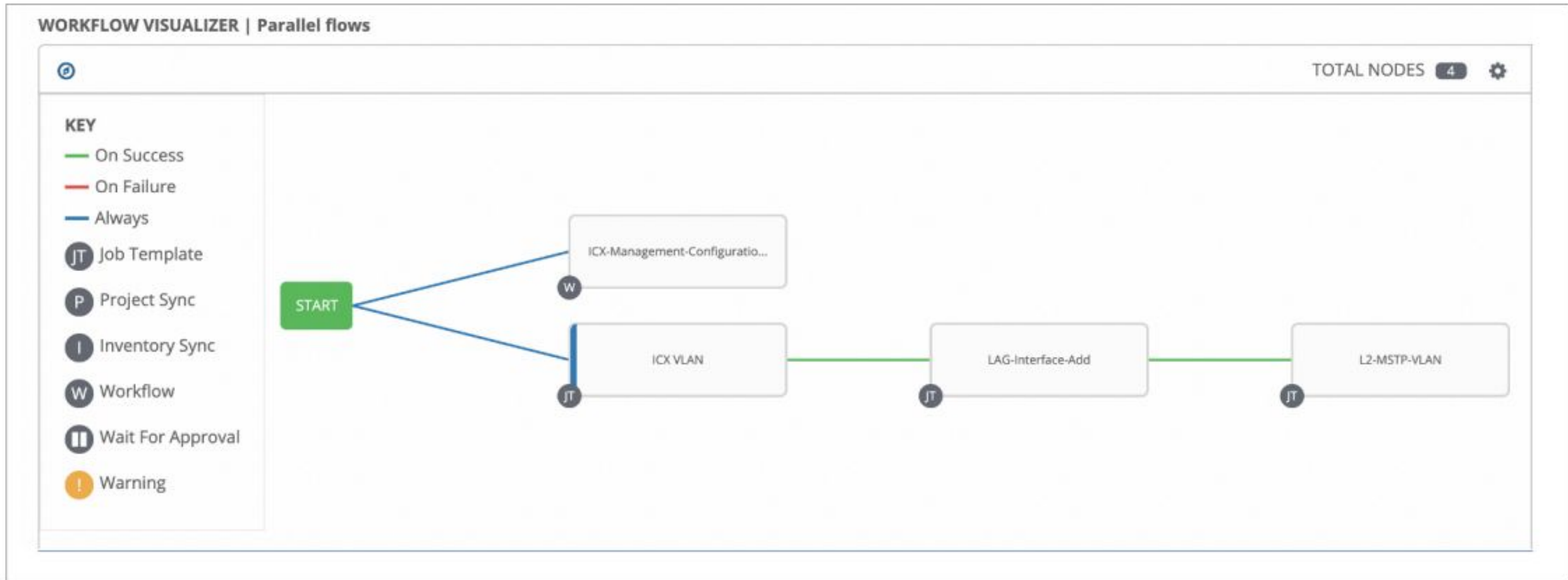
67 awxserver : ok=16 changed=

0 unreachable=0 failed=0 skipped=1

68 rescued=0 ignored=0

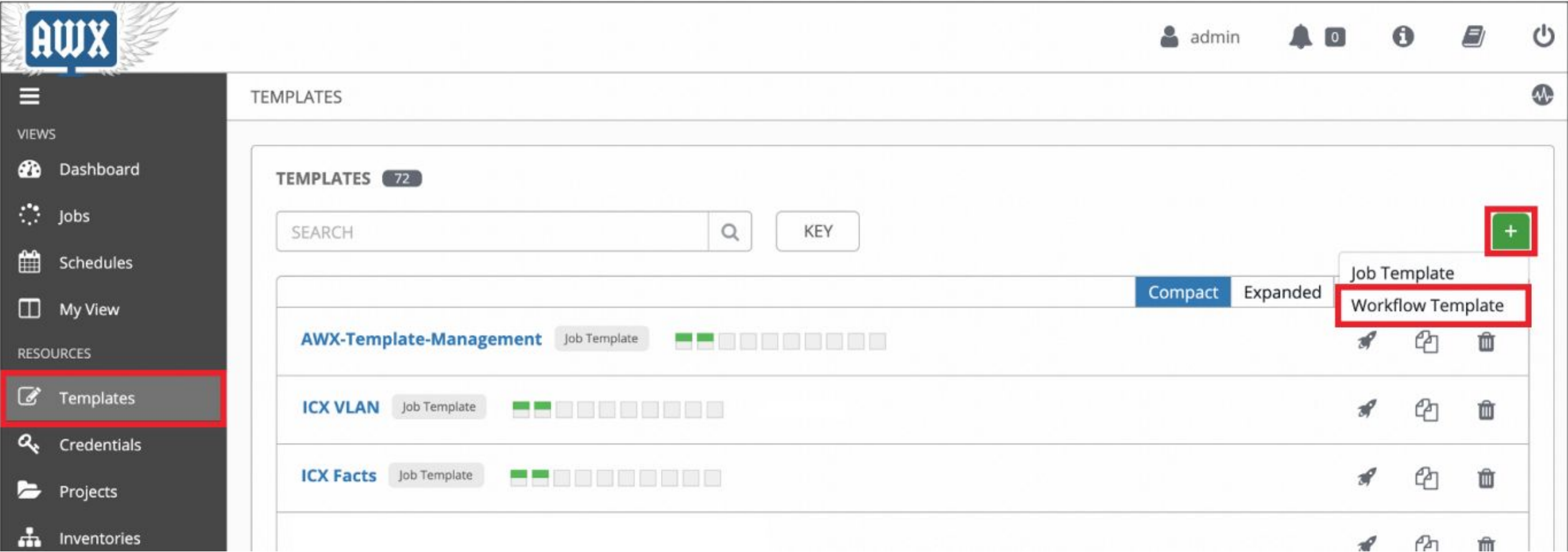
Workflow Templates

Ansible AWX allows the creation of workflow templates, where several job templates can be tied together in a workflow, executing tasks in sequence or in parallel:



CREATING A WORKFLOW TEMPLATE

Click on the Templates button, followed by the + button, then click on Workflow Template:



CREATING A WORKFLOW TEMPLATE

The Create Workflow template will open. Complete the Name and Inventory fields:

NEW WORKFLOW JOB TEMPLATE

DETAILS

PERMISSIONS

COMPLETED JOBS

SCHEDULES

ADD SURVEY

WORKFLOW VISUALIZER

* NAME

ICX-workflow

DESCRIPTION

ORGANIZATION

Q

INVENTORY ?

Q

ICX Switches

PROMPT ON LAUNCH

☐

LIMIT ?

PROMPT ON LAUNCH

☐

SCM BRANCH ?

PROMPT ON LAUNCH

☐

LABELS ?

OPTIONS

☐ ENABLE CONCURRENT JOBS ?

☐ ENABLE WEBHOOK ?

EXTRA VARIABLES ?

YAML

JSON

PROMPT ON LAUNCH

☐

1

LAUNCH

CANCEL

SAVE

CREATING A WORKFLOW TEMPLATE

Click on the START node. A new empty node will appear. On the right panel you will find the available templates (including job templates and workflow templates):

The screenshot displays the 'WORKFLOW VISUALIZER | ICX-workflow' interface. The main workspace shows a workflow with a green 'START' node connected to a dashed rectangular placeholder node. A red box highlights the 'START' node. On the right panel, a list of templates is shown, with 'show LLDP neighbor' selected and highlighted by a red box. Below the list, there are pagination controls (PAGE 6 OF 8, ITEMS 51 - 60 OF 74) and configuration options for 'RUN' (set to 'Always') and 'CONVERGENCE' (set to 'Any'). At the bottom right of the right panel, there are 'CANCEL' and 'SELECT' buttons, with the 'SELECT' button highlighted by a red box.

WORKFLOW VISUALIZER | ICX-workflow

TOTAL NODES 1

START

show LLDP neighbor

Remote-User-LocalAcct

Remote-User-StrictMode

System-Misc-Banner

System-Misc-ConsoleHarden

System-Misc-DNS

« < 4 5 6 7 8 > » PAGE 6 OF 8 ITEMS 51 - 60 OF 74

* RUN

Always

* CONVERGENCE

Any

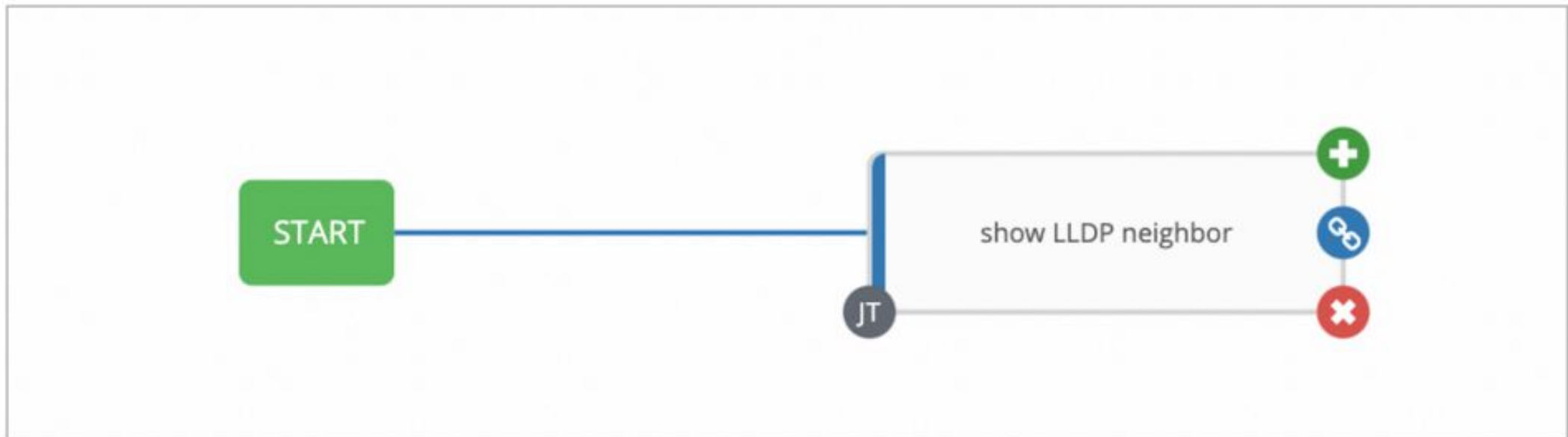
The inventory of this node will not be overridden by the parent workflow inventory.

CANCEL SELECT

CREATING A WORKFLOW TEMPLATE

When you select the template, it gets installed in the empty node in the workflow. Click on the second node. Three icons will appear along the node right border:

- The + icon is used to add another node to the flow (either in the same branch or on a new branch)
- The link icon is used to create a link to connect the node to any available node
- The x icon is used to delete the node



LAUNCHING THE WORKFLOW

When you select the template, it gets installed in the empty node in the workflow. Click on the second node. Three

ICX-workflow

DETAILS

PERMISSIONS

NOTIFICATIONS

COMPLETED JOBS

SCHEDULES

ADD SURVEY

WORKFLOW VISUALIZER

★ NAME

ICX-workflow

DESCRIPTION

ORGANIZATION

Q

INVENTORY ?

Q

ICX Switches

PROMPT ON LAUNCH

☐

LIMIT ?

PROMPT ON LAUNCH

☐

SCM BRANCH ?

PROMPT ON LAUNCH

☐

LABELS ?

OPTIONS

☐ ENABLE CONCURRENT JOBS ?

☐ ENABLE WEBHOOK ?

EXTRA VARIABLES ?

YAML

JSON

PROMPT ON LAUNCH

☐

1 ---

LAUNCH

CANCEL

SAVE

LAUNCHING THE WORKFLOW

Each node will execute (in sequence or in parallel). A green dot at the node upper left corner indicates the job template executed correctly. When all nodes complete the execution, you should see a Successful status on the left pane.

The screenshot displays a workflow execution interface. On the left, a 'DETAILS' panel shows the status as 'Successful', which is highlighted with a red box. The status is preceded by a green dot. Other details include: STARTED (5/10/2021 7:43:07 PM), FINISHED (5/10/2021 7:43:30 PM), INVENTORY (ICX Switches), TEMPLATE (ICX-workflow), and LAUNCHED BY (admin). Below these details is an 'EXTRA VARIABLES' section with tabs for 'YAML', 'JSON', and 'EXPAND'. The 'YAML' tab is selected, showing a single variable '1 ---'. On the right, the 'ICX-workflow' diagram is shown. It features a blue square start node connected to a green rectangular node labeled 'show LLDP neighbor', which is then connected to another green rectangular node labeled 'ICX-Edge Port'. Both nodes have a green dot in their top-left corner, indicating successful execution. Each node has a 'JT' icon and a 'DETAILS' link below it. At the top right of the workflow panel, it shows 'TOTAL NODES 2' and 'ELAPSED 00:00:22'.

DETAILS

STATUS **Successful**

STARTED 5/10/2021 7:43:07 PM

FINISHED 5/10/2021 7:43:30 PM

INVENTORY [ICX Switches](#)

TEMPLATE [ICX-workflow](#)

LAUNCHED BY [admin](#)

EXTRA VARIABLES ? [YAML](#) [JSON](#) [EXPAND](#)

1 ---

ICX-workflow TOTAL NODES 2 ELAPSED 00:00:22

show LLDP neighbor

ICX-Edge Port

LAUNCHING THE WORKFLOW

To see the execution details, click on DETAILS inside the node.

DETAILS

STATUS ● Successful

STARTED 5/10/2021 7:43:08 PM

FINISHED 5/10/2021 7:43:17 PM

JOB TEMPLATE [show LLDP neighbor](#)

JOB TYPE Run

LAUNCHED BY [admin](#)

INVENTORY [ICX Switches](#)

PROJECT [ICX Test](#)

PLAYBOOK [icx_cli.yml](#)

CREDENTIAL [ICX Login](#)

ENVIRONMENT [/var/lib/awx/venv/ansible](#)

EXECUTION NODE [awx](#)

INSTANCE GROUP [tower](#)

SOURCE [ICX-workflow](#)

WORKFLOW

EXTRA VARIABLES [YAML](#) [JSON](#) [EXPAND](#)

1 ---

show LLDP neighbor

PLAYS 1 TASKS 3 HOSTS 1 ELAPSED 00:00:09

SEARCH Q KEY

```
*****
13 ok: [ICX 7150 - Marcelo] => {
14     "msg": {
15         "changed": false,
16         "failed": false,
17         "stdout": [
18             "Lcl Port Chassis ID      Port ID
Port Description      System Name  \n1/1/
1  b479.c80d.7630 b479.c80d.7630 eth0
H510-A"
19         ],
20         "stdout_lines": [
21             [
22                 "Lcl Port Chassis ID      Port
ID      Port Description      System N
ame ",
23                 "1/1/1      b479.c80d.7630 b47
9.c80d.7630 eth0      H51
0-A"
24             ]
25         ]
26     },
```

Role-Based Access Control (RBAC)

In the *Ansible AWX* web interface, users need different levels of access based on their roles.

- ✓ Some users may only need to execute existing job templates on a predefined inventory of machines.
- ✓ Others may require permissions to modify specific inventories, job templates, or playbooks.
- ✓ Certain users, such as administrators, need full control to make changes across the entire AWX configuration.

By default, AWX includes a built-in administrative user, **admin**, with superuser privileges, granting full access to the system configuration.

- ✓ User permissions for managing AWX objects are controlled through Role-Based Access Controls (RBAC). RBAC assigns roles to users, defining their permissions and specifying exactly who can view, modify, or delete objects within the AWX web interface.

Role-Based Access Control (RBAC)

Ansible AWX provides Role-Based Access Control (RBAC) to manage these access levels. For instance:

- ✓ You may want some users to only view job templates.
- ✓ Others might need to modify playbooks or inventories.
- ✓ Some may only have read-only access to the Ansible AWX.

The RBAC system allows you to assign specific roles to users, ensuring they only have access to the resources they need.

Role-Based Access Control (RBAC)

- **User Types**

- ✓ By default, the Ansible AWX installer creates an administrative user account, **admin**, which has full control over the AWX Interface.

- ❑ Additional users in AWX can be created under one of the following three user types:

- ✓ System Administrator

- ✓ System Auditor

- ✓ Normal User

Role-Based Access Control (RBAC)

- **ORGANIZATION**

- ✓ An Ansible AWX Organization is a logical grouping of Teams, Projects, and Inventories. Every user must be assigned to an Organization, making it essential to understand Organizations before diving into user configuration.

- **ORGANIZATION ROLES**

- ✓ Newly created users inherit roles from their assigned Organization based on their user type.
- ✓ Additional roles can be granted to users after their creation to provide permissions for viewing, using, or modifying additional AWX objects.

❑ The available Organization roles include:

- ✓ Admin ✓ Member
- ✓ Auditor ✓ Read

Thank you

QUESTIONS?